Victorian 6502 User Group Newsletter

# KAOS

For People Who Have Got Smart



OSI SYM KIM AIM APPLE UK101 ORANGE

Registered by Australia Post
Publication No. VBG4212

ADDRESS ALL CORRESPONDENCE TO 10 FORBES ST., ESSENDON, VIC. 3040

Vol.3 No.1 October 1982

After all the rumours that have been flying around for the last few months it is finally official, Macom/OSI are going out of the personal computer business. Since contacting Macom to confirm this, we have been doing some research and have found that very little hardware or software was coming from OSI anyway and that quite a few companies in America are now specializing in material for the OSI personal computer and they seem to think that there are enough OSIs to keep them in business for quite some time. When we spoke to the publisher of PEEK (65) he was quite confident that his 2000 subscribers would be receiving PEEK for a long time to come. As you will see from the letter on page 2, the publishers of MICRO also intend to be publishing articles on the OSI for some time yet.

Australia is producing some very good hardware, the Tasker Buss and Video board, the Rabble Ozi Expansion board, the 48K CMOS RAM board and numerous small boards produced by David Anear and David Tasker whose next project will be the 16 pin I/O buss and all the goodies that will plug into it. Also, from Shepparton, the new OSI compatable single-board computer, which should be available early next year, will be an up-graded replacement for the Superboard.

Some very good OSI software is being written in Australia, for example the new Assembler/Editor written by Louis Madon for COMP-SOFT and the educational software which is starting to appear from our schools as teachers realize that they can write programs that are equal to or better than any from overseas.

So to summarize, after some initial wailing and gnashing of teeth, people are starting to realize that for the OSI personal computer it will be business as usual and the future is looking bright.

The next meeting will be on Sunday 31st October 1982 at 2pm at the Essendon Primary School which is on the corner of Raleigh and Nicholson Streets, Essendon.

The closing date for items for the next issue is 12th November

**********

INDEX

## PEACH MENU SELECTION MADE EASY

*by Erik Sundstrup*

Loading and running programmes from the menu soon becomes tiresome with all the typing necessary. I thought there had to be a better way, so I devised this programme which makes file selection a breeze.

I make my programme the destination programme after a disk auto-boot. The familiar files appear but also a little indicator beside the file names. Simply position this with the up and down arrows to the selected file and hit RETURN. The file will now load and auto-run, be it in Basic or Machine Code. There is no response to DATA files, invalid key entries or a mis-positioned cursor. Entering Q will exit the program.

The INPUT$ command allows writing the keyboard to memory without the screen, except when asked. The CSRLIN statement allows the name of the file (beside the indicator) to be READ and used for the RUN command. This name is read from screen memory and the eleventh bit is used to sense Basic, Machine or Data files. Only the first 8 bits are used for the RUN command.

```
10 CONSOLE 0,25 0:CLS:WIDTH40:COLOR7,0:FILES:ON ERROR GOTO 80:R$=STRING$(11,53)
20 LOCATE 8,0,1:PRINT"<>"
30 X$=INPUT$(1):IF X$=CHR$(30) OR X$=CHR$(31) THEN PRINT X$;:Z=CSRLIN:IF Z<1
    THEN Z=1
40 IF X$=CHR$(31)THEN LOCATE8,Z-2:PRINT SPC(2):LOCATE 8,Z-1:PRINT"<>":GOTO30
50 IF X$=CHR$(30) THEN LOCATE 8,Z:PRINT SPC(2):LOCATE8,Z-1:PRINT"<>":GOTO30
60 IF X$=CHR$(13)THEN IF PEEK(1000+Z*40)<1 THEN 30 ELSE R$="":FOR T=0TO10:
    R$=R$+CHR$(PEEK(984+T+Z*40)):NEXT T
65 IF X$="Q" THEN CLS:LOCATE 0,0,3:END: REM Allows you to break out of program
70 V=VAL(MID$(R$,11,1)):D$=LEFT$(R$,8):IF V=1 THEN 30 ELSE IF V=0 THEN CLS:
    LOCATE 0,0,3:LOAD D$,R:ELSE IF V=2 THEN LOCATE 0,0,3:CLS:LOADM D$,,R ELSE
    30
80 Z=24: RESUME 40
```

**********

EXTRACT of a LETTER to KERRY LOURASH from MICRO INK, Inc.

*Dear Sir,*

*We have communicated with OSI and they have assured us that they are not going out of business. However they are going out of the personal computer business and only will be making business computers. We at MICRO do feel an obligation to our loyal OSI readers and we will continue to provide coverage of OSI personal computers. We are concerned about the quantity of OSI material that we are able to present, compared with the backlog of OSI articles in our article queue.*

*We are thinking strongly about producing an OSI book to be a technical resource for OSI owners. We would appreciate input from you OSI people about your response to this idea. Also we would like input about what specific articles would be of most interest to you, so that we would know which articles you would like to see published.*

*Please let us know your thoughts on these matters. Looking to hear from you. Thank you for your concern.*

*Sincerely,*
*Phil Daley, Editor*

# LOW COST LOGIC PROBE

*by David Anear*

I am frequently rung up by people who have problems with their computer or a board they are making. I don't mind this, but it is very difficult to make repairs over the phone when the person calling doesn't have a logic probe to check clock pulses etc.

I realize that most people are reluctant to pay $20 to $200 for something they may only use once or twice a year, but this little device is essential if you are going to attempt repairs or modifications on digital circuits. Incidently a logic probe is far better than a CRO (which even fewer people have access to) for detecting fast pulses at a slow repetition rate.

Over the last few months, I have been looking around at various probe designs to find one that is not, too expensive, too elaborate, or requires selected or matched components. The one I have chosen was originally published in Practical Electronics in 1977 and fills all the requirements, it is simple, easy to construct and cheap, -it should cost $3 to $4 to make.

The probe uses 2 LEDs to indicate the state of the signal being tested

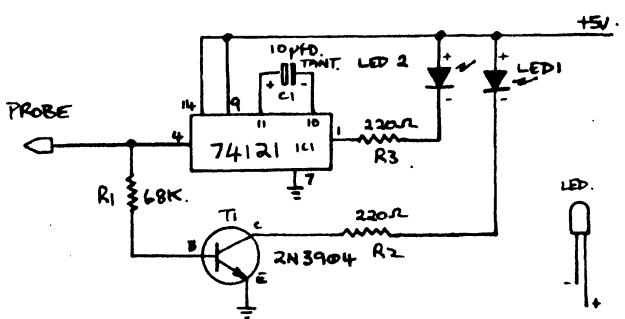
LED 1: With open circuit point or no signal applied this LED glows dimly, at low logic level or earth it goes out. At high logic level or +5V the LED glows brightly.

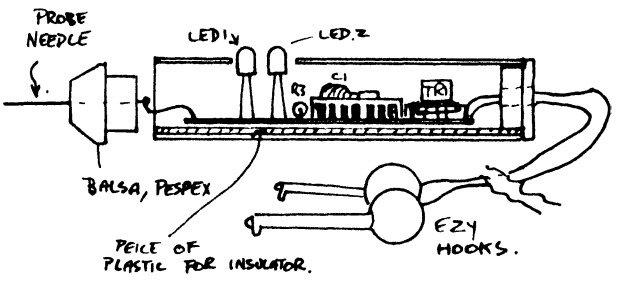
LED 2: This is the pulse LED and it flashes as the logic level changes from High to Low, as the pulse rate increases the LED will flash more frequently until it appears to be on continuously.

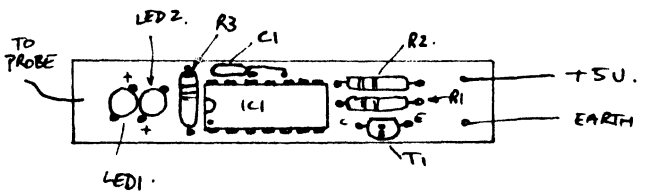
| SIGNAL | LED1 | LED2 |
|---|---|---|
| Hi or +5V | Bright | Off |
| Open circuit | Dull | Off |
| Low or Earth | Off | Off |
| Pulsing | On | On |

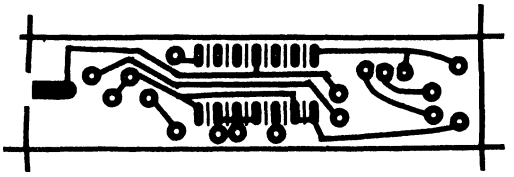
The circuit is powered from the unit under test and is connected to +5V and earth with Ezy Hooks or alligator clips. If you connect the power leads the wrong way round you have approximatly .001 seconds to disconnect the leads before the probe expires, a good chance to test your reflexes.

The circuit can be built on a printed circuit board or a piece of Vero board. I used a couple of pieces of aluminium 'U' section to make a case but anything you have handy will do. The main thing is to have a fine, strong, rigidly mounted point so that you can push the probe firmly against an IC pin without the point bending or slipping and touching another pin.



# Superboard

NEWSLETTER OF THE OHIO SUPERBOARD USER GROUP, 146 YORK STREET, NUNDAH, 4012.

## PHONEME TEST PROGRAM by Ray Richards

```
 10 FOR S=5120 TO 5126:READ M:POKE S,M:NEXT:POKE 11,0:POKE 12,20
 20 DIM T$(63),L(100):X=1:FOR A=0 TO 63:READ T$(A):NEXT
 30 Y=0:INPUT"PHONEME";X$:IF X$="PC"THEN 140
 40 IF X$ ="STOP"THEN 80
 50 IF T$(Y)=X$ THEN L(X)=Y+I:X=X+1:GOTO 30
 60 Y=Y+1:IF Y>63 THEN PRINT"INVALID PHONEME":GOTO 30
 70 GOTO 50
 80 L(X)=63:R=X:X=1
 90 FOR X=1 TO R:POKE 535,L(X):X=USR(X):NEXT
100 INPUT"LISTEN AGAIN";A$:IF MID$(A$,1)="Y"THEN 90
110 PRINT:PRINT:FOR S=1 TO R:PRINT L(S);:NEXT
120 FOR D=1 TO 10000:NEXT:PRINT:PRINT
130 PRINT"SELECT NEW WORD":PRINT:X=1:I=0:GOTO 30
140 INPUT"INTONATION";I:IF I>2 THEN 140
150 IF I=1 THEN I=64
160 IF I=2 THEN I=128
170 GOTO 30
180 DATA 17,3,23,2,32,177,252,96
190 DATA EH3,EH2,EH1,PA0,DT,A2,A1,ZH,AH2,I3,I2,I1,M,N,B,V
200 DATA CH,SH,Z,AW1,NG,AH1,OO1,OO,L,K,I,H,G,F,D,S
210 DATA A,AY,Y1,UH3,AH,P,O,I,U,Y,T,R,E,W,AE,AE1
220 DATA AW2,UH2,UH1,UH,O2,O1,IU,U1,THV,TH,ER,EH,E1,AW,PA1,STOP
```

## 3D PLOT by Marty Vlekkert

This program suits 64 character screens or a line printer. In the case of a printer, get into the SAVE mode before running the program.

```
 10 DEF FNA(Z)=30*EXP(-Z*Z/100)
 20 FOR X=-30 TO 30 STEP 1.5:L=0:Y1=5*INT(SQR(900-X*X)/5)
 30 FOR Y=Y1 TO -Y1 STEP-4:Z=INT(25+FNA(SQR(X*X+Y*Y))-.7*Y)
 40 IF Z<=L THEN 60
 50 L=Z:PRINT TAB(Z);"*";
 60 NEXT Y:PRINT:NEXT X:END
```

## MODEMS

We have a number of RACAL direct connect telephone modems available through the personal efforts of one of our most active Brisbane members.

These are fully professional send/receive units using Australian frequency standards. The modem consists of a 190mm x 150mm printed circuit board, a DB25 socket, a power transformer and two fuses in a 275mm x 170mm x 55mm diecast aluminium box.

The modems are at reasonable cost. Unfortunately, we do not have enough to go to all OSUG members, so if you believe that you could productively use one, you are invited to put your reasons in writing to me. Please enclose an SAE with your letter. STD costs make modems an expensive exercise compared with cassette tapes for country members. The modems are in excellent condition.

# Superboard

## SOFTWARE REVIEW - Escape from Mars

Escape from Mars is an 8k adventure. Adventures are fantasy role-playing games. The program creates and runs a fantasy world where the player has the role of some character who has some task to complete. Sometimes even the task is unknown. In Escape from Mars, the object of the game is to refuel your rocket and take off.
Using two word commands like "take key" (ta ke will do), you tell your character what to do and where to go. The program then takes appropriate action and transports your character to various locations and describes them.
As in all unfamiliar situations, great attention should be paid to objects and messages, because what you don't notice can kill your character, and then you lose the game.
Adventures are the ultimate in puzzles and can take thirty hours or more to solve. You need a maximum of imagination and determination, because much of the normal wordy stuff and clues have to be omitted to enable an OSI adventure to fit into 8k. Escape from Mars is not a particularly hard-to-solve adventure.

Escape from Mars comes as two tapes, one being a software fix for the faulty garbage collector routine in Basic 3. If you have a re-programmed Basic 3, remove all the calls to USR(X) in the main program and do not load the garbage collector tape. Escape from Mars is available from Aardvark Technical Services, 2352 South Commerce, Walled Lake, MI 48088, U.S.A.
The OSUG library has the program. Postage required 55¢ + 27¢ stamps + label.

---

## EXTENDED BASIC COMMANDS

Extended Basic Commands is a machine code program occupying $1800 - 1FFF. On loading, it self runs, and patches in some additional commands as if they were in the Basic ROMs. The main function of the program is as a utility to assist with debugging and tidying up Basic programs.

DELETE : Deletes any number of lines from your program.
COPY : Copies the contents from one line number to another.
TRACE : Enables bugs to be found with ease.
RENUMBER : A fast line renumberer. Any starting line, any increment.
CHANGE : The ultimate editor. Will hunt through your program for commands, variables, numbers or strings and change them to anything you desire subject to your final approval. Corrects spelling errors, shortens or expands lines or text. Delete part of a line.

The code is self modifying but can be relocated to any part of RAM using an Extended Monitor program and a list of changes which can be provided. E.B.C. comes on a C10 cassette and is suitable for the C1P. You need to specify Synmon or Cegmon monitor version. Also works with Dabug. Cost is $6.80.

---

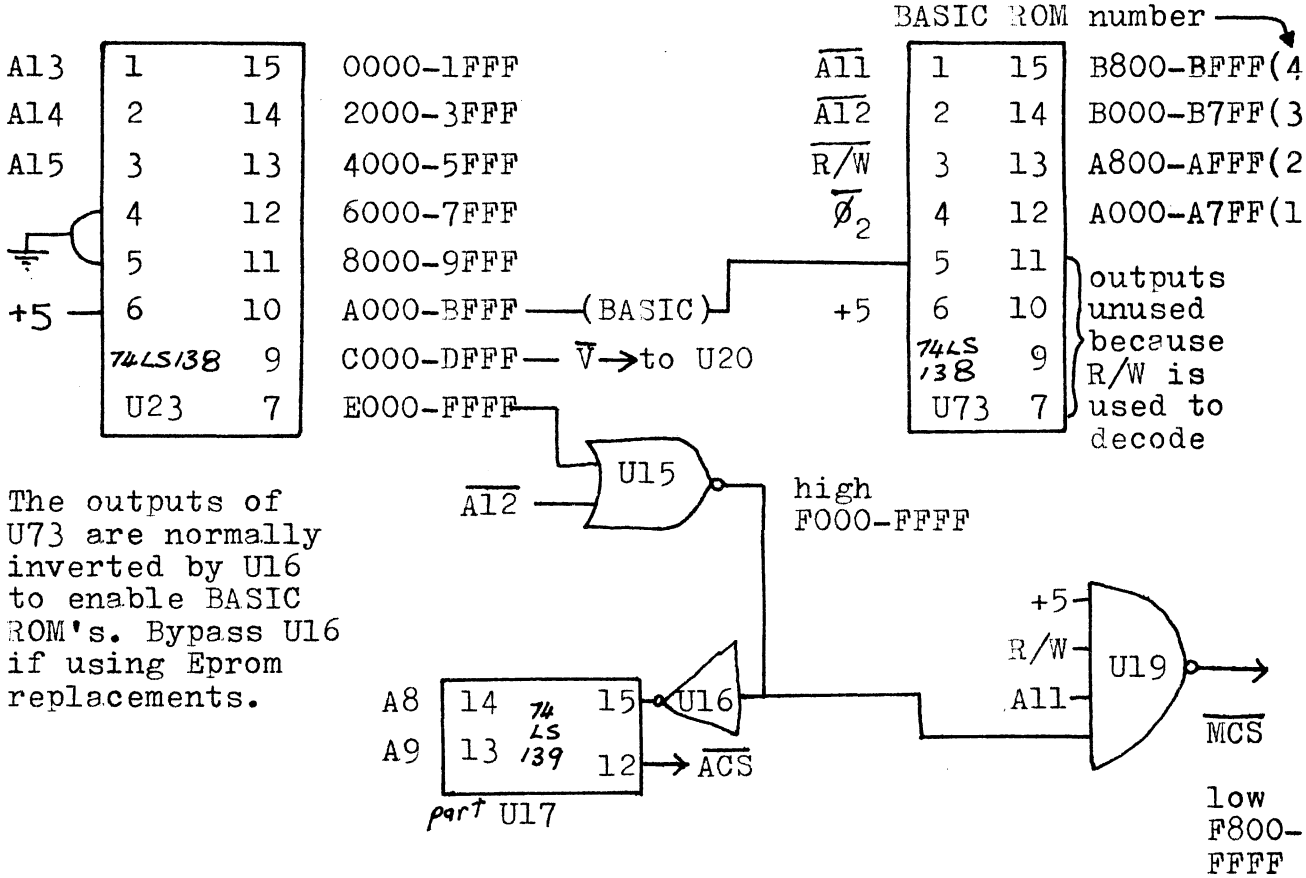
## MORE INPUT PLEASE

I was sorry to see an article in the last KAOS newsletter regarding the lack of input from members. Indeed, SUPERBOARD appears here because of a similar problem. Unfortunately, as in most non-profit organisations, a few hard-working members keep the whole thing together. However I was heartened by a thoughtful article on items which might be of interest, and will follow this up and see what I can provide. I would like to hear more about what the "silent majority" would like to read about in their newsletter. Next month, SUPERBOARD will feature an inexpensive, fast access cartridge tape system.

*Ed Richardson.*

ADDING EPROMS TO THE SERIES II SUPERBOARD.

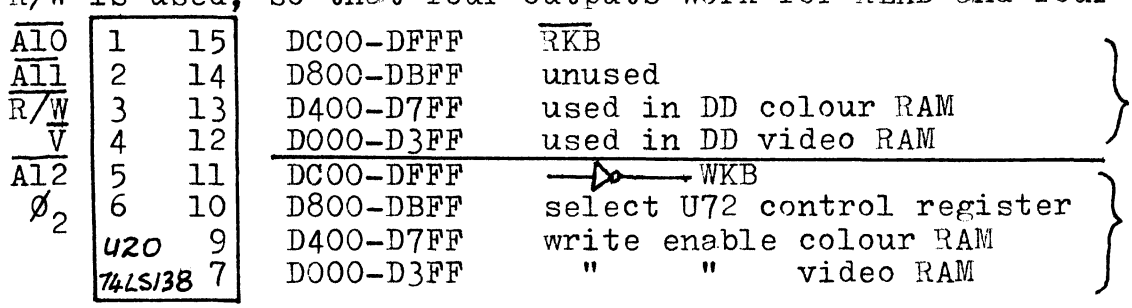
The need to have a close look at address decoding started after it was decided to build an Eprom programmer, which needed to be put somewhere in the memory map itself, and it also offered the prospect of adding Exmon to the Superboard. More about the Eprom programmer later, but first a bit about addressing. The first diagram shows how the standard decoding in the Series II looks. The main difference in Series I is that there is no U73, and the BASIC ROM's are selected by the other half of U17 (74LS139), which is unused in Series II.




The outputs of U73 are normally inverted by U16 to enable BASIC ROM's. Bypass U16 if using Eprom replacements.

While the ACS signal is low from F000-F0FF, A11 and A10 have not been used in the decoding, and so this signal will also be low anywhere else between F000-FFFF where A8 and A9 are both low. (F400-F4FF ; F800-F8FF ; FC00-FCFF ). This killed an idea I tried, using $\overline{ACS}$ instead of A11 to U19 as above. This would have enabled a 2732 Eprom from F100-FFFF, allowing F800-FFFF for the monitor, and F100-F7FF for other machine code routines.

Some of the U20 outputs below are not used in Series I . Again, R/W is used, so that four outputs work for READ and four WRITE.



Using D800-DBFF. The unused output on pin 14 of U20 goes low if an address in the range 55296-56319 is read, such as with a PEEK command. This can be used in conjunction with a 7474 to control another function, such as an alternative character generator in a 2716 or 2732. Same applies to Series I.

BASIC and Eproms. Replacement of some OSI ROMs started with BASIC 3 to fix the string bug, followed by Ed Richardson's BASIC 4, which has proved extra good for the rapid token save and filename option. A mod to BASIC 1 from OSUG newsletter 21 has allowed access to graphics beyond $80 via the Repeat key. By this stage I had found that the decoding of U73 was wasting outputs which could be used for add-ons such as Exmon. The left diagram shows how to change U73 to give eight 2K blocks. If you take pin 18 of Eproms to earth, and use pin 20 to select, the system will work at 2 MHz. The OSI ROMs have pin 18 at +5 and pin 20 high to select. I left BASIC 2 where it was and put a socket on top for BASIC 1, bearing in mind that pins 18 and 20 are different. BASIC 4 was put on top of 3, which was fairly easy as both were in 2716. There were now two free sockets for other 2716 or 2732 Eproms .

If you prefer not to piggyback the BASIC chips and can put them in 2732 Eproms, the right hand diagram shows how to get eight 4K blocks. I haven't tried this version, but the 2K decoding with piggybacked BASICs is working fine, as is the extended monitor at 8800-8FFF. In both diagrams, pins 2 and 4 are as original. A13 was taken from the 6502 and inverted with an unused section of U16. Cut R/W to U73 pin 3. Cut track from U23 pin 10 (was BASIC block select) and join A14 from 6502 to U73 pin 5.(For the 4K version, invert A14 and take to U73 pin 1). In both cases, A15 is taken from 6502 to pin 6 of U73.

| | | | | | |
|---|---|---|---|---|---|
| $\overline{A11}$ | 1 | 15 | B800-BFFF | Basic 4 | 2K blocks |
| $\overline{A12}$ | 2 | 14 | B000-B7FF | Basic 3 | |
| $\overline{A13}$ | 3 | 13 | A800-AFFF | Basic 2 | |
| $\emptyset_2$ | 4 | 12 | A000-A7FF | Basic 1 | |
| A14 | 5 | 11 | 9800-9FFF | ? | |
| A15 | 6 | 10 | 9000-97FF | Basic 5 ? | |
| | | 9 | 8800-8FFF | Exmon ? | |
| U73 | | 7 | 8000-87FF | Toolkit ? | |

4K blocks

| | | | |
|---|---|---|---|
| $\overline{A14}$ | 1 | 15 | F000-FFFF |
| $\overline{A12}$ | 2 | 14 | B000-BFFF |
| $\overline{A13}$ | 3 | 13 | E000-EFFF |
| $\emptyset_2$ | 4 | 12 | A000-AFFF |
| earth | 5 | 11 | D000-DFFF |
| A15 | 6 | 10 | 9000-9FFF |
| | | 9 | C000-CFFF |
| U73 | | 7 | 8000-8FFF |

You can retain the BASIC in 2K chips (EPROM or ROM) and put any extra programs in 2732 Eproms by using the unused section of U17. This might be the choice if you don't want to change U73 as outlined above, but can program 2732's. I have tried this and it is OK at 2MHz if the 2732 has pin 18 to earth, and pin 20 is used to select. Join pin 11 U23 to pin 1 of U17. Join $\emptyset_2$ from U73 pin 4 to U17 pin 2. Join $\overline{A12}$ from U73 pin 2 to pin 3 of U17. (no cuts). Pin 6 U17 is low 8000-8FFF and 4 is low 9000-9FFF.

EPROM PROGRAMMER.
This programmer suits 2716/2732 Eproms, and uses two VIA's which are decoded in KAOS suggested format of C030-C03F and C040-C04F, and nowhere else. It is built on vero board and hooked to the Superboard with a 40-way cable, but could be made to plug into the Tasker bus with some wiring jumpers. Data and address lines are buffered. Works only at 1 MHz, and needs about +28V to the on-board 317 regulator. BASIC software is available (3.7K), thus allowing a 2732 to be programmed with an 8K machine. For a copy of circuits and notes (5 pages) and a cassette dump of program, send six stamps (current letter rate) and cassette (no case) to Berrie Hills, If interested, write soon, not in 1984.

## GET THOSE M/C CASSETTE PROGRAMS ONTO DISK

*by Jeff Kerry*

Upgraded to disk, or thinking about it? (Does 2.5 seconds to load a program sound more interesting than 7. minutes?) Here's a way to get machine-code cassette programs onto a disk in self-loading, auto-starting format, almost regardless of their normal memory location.

Many machine code programs for the C1 occupy the same memory area as part of Disk Basic (hex $0200 up), and may be difficult or impossible to relocate to higher memory space. This means you can't call them directly in from disk using a Basic program, since the call would clobber Basic. The answer is to call them, track by track, into a higher memory area, (eg. the $4000's), then Execute a blockmove routine, designed to shift your program down to where it was originally supposed to go, and jump to its entry address.

This means that all you need to do is to RUN the simple Basic program and presto! in a couple of seconds, there's your machine code program in and running.

The example given is for an early Aardvark Chess program. Running the original checksum format cassette reveals that it commences at hex $0266 and ends at about $0AFF, and has an entry point at $03AC. It also uses page zero addresses (from $0000 to $00FF.)

STEP 1: Boot the disk you wish to use. Check its DiRectory to see what tracks are available. This example uses 4 tracks: 21 & 22 for the machine code data "CHESDA", 23 for the Basic program "CHESS" and 24 for the block move routine "CHSMOV". If you do not have COMP-DOS you will need to CReate these files now. If you do, just enter the following:

```
DISK!"CR CHESDA,21,22"
DISK!"ZE CHESDA"
```

STEP 2: Type (Break) M L and load the cassette program.

STEP 3: Type (Break) M 2A51 G (to get back into DOS)

```
SA 21,1=0266/8  (the 1st 8 pages)
SA 22,1=0A66/1  (the 9th page)
SA 22,2=0000/1  (the page zero tables).
```

STEP 4: Now the Basic program. Reboot the disk and type NEW

```
10 REM  CHESS EXEC.
20 REM
30 DISK!"CA 4266=21,1"
40 DISK!"CA 4A66=22,1"
50 DISK!"CA 0000=22,2"
60 DISK!"XQ CHSMOV"
```

Save it: DISK!"PU CHESS"

STEP 5: Finally the block move routine. This is best entered in Assembler format, so type EXIT and AS :

```
10 ;     CHSMOV     BLOCK MOVE ROUTINE
20 ;
30    *=$327E ; OCCUPIES NORMAL WORKSPACE BUFFER
40  CHESS = $0266
50  HIGH  = $4266
60  ENTER = $03AC
70 ;
80          LDA #0
90          CLI
100         TAX         ;ZERO X REGISTER
110         TAY         ;ZERO Y REGISTER
120  PAGE   LDA HIGH,Y
130  STORE  STA CHESS,Y
140         INY
150         BNE PAGE    ;MOVE 1st PAGE
160         INC PAGE+2  ;NEXT PAGE
170         INC STORE+2
180         INX
190         CPX #09 ;9 PAGES IN THIS EXAMPLE
200         BNE PAGE
210         JMP ENTER   ;GO AND START
```

Type A to assemble it and check for errors. You may want to Save a copy of this before you assemble it to memory, since it clobbers itself, so type !PU MOVER. Finally, type A3 and then !PU CHSMOV .
That's all. Reboot the disk and type RUN"CHESS" . These techniques should work satisfactorily on most machine code cassette programs.

**********

## THE BEGINNING MACHINE LANGUAGE PROGRAMMER part 4

*by David Dodds*

Although the 6502 has 56 different instructions many are so similar that they are easily learned as a group. The operations LOAD, STORE, INCREMENT, DECREMENT and BRANCH are good examples of this similarity.

The LOAD instructions all involve fetching imformation from memory and then placing it in one of the registers. There are many ways of specifying which memory location the imformation is to be obtained from (using the different addressing modes). However the basic principle is the same. The imformation may be loaded into the accumulator or into either of the index registers. Thus we have available LDA, LDX and LDY (load accumulator, x register and y register respectively). During the fetching operation the CPU tests whether the value is zero, negative ()$7F) or positive and sets the status flags N and Z according to the value. Eg when the instruction LDA $1000 (absolute mode) is performed if $1000 contained $20 then N will be set false (0) and Z also will be false. The settings of the other flags remain unchanged.

STORE is the opposite of load. The contents of the accumulator, or of either of the index registers is put into memory. These operations are known as STA, STX and STY. The store instructions do not affect the status flags.

INCREMENT instructions add one to the value in an index register or in memory; ie like X=X+1 in BASIC.

DECREMENT subtracts one from the value in a register or memory location. The accumulator can not utilize the increment and decrement instructions. The mnemonics for the increment instructions are INC (memory), INX (x reg.) and INY (y reg.). Similarly for decrement they are DEC, DEX and DEY. The CPU tests the value after it is modified by one of these instructions and sets the N and Z flags to show the status. Eg if X=$7F and an INX is performed then X will be $80 (-128 two's complement notation), Z will be 0 (false) and N, 1 (true). All other flags are unchanged.

The function of the BRANCH instruction has been covered previously but to recap:

-branch instructions are conditional ie like an IF...THEN GOTO
-the condition test is carried out on the status flags
there are 8 branch instructions known as:

```
BEQ  Branch if equal zero  (Z=1)
BNE    "    if not equal zero  (Z=0)
BMI  Branch if minus  (N=1)
BPL    "    if plus  (N=0)
BCS  Branch if carry set  (C=1)
BCC    "    if carry clear  (C=0)
BVS  Branch if overflow set  (V=1)
BVC    "    if overflow clear  (V=0)
```

-if the test did not meet the condition specified for a branch the program continues execution sequentially, otherwise the byte following the branch opcode is used to calculate a new value for the program counter.
-The range of a branch is +129 to -126 relative to the location of the branch opcode.
-The branch instructions do not affect the status register.

*Next page please*

With what we have learned so far it is possible to start writing programs. The one I have selected for demonstration is a screen clear.

A screen clear routine loads screen memory with the ASCII character $20, a blank. As we only know absolute and immediate mode instructions we will need to use what is known as self modifying code, ie the program will change memory where the program itself is stored.

The first thing needed is a pointer to the location which is to be cleared. The pointer will be the address part of an STA absolute instruction. This instruction will form the heart of the program. The first thing our program must do is initialise the pointer. If we call the place where the STA instruction is located TVPUT, then the pointer will be at memory location TVPUT+1 and TVPUT+2.

Here the first peculiarity of the 6502 appears. Absolute addresses are stored in memory in the order low byte, high byte which is the opposite of the way they are written in a program. When setting up the pointer to the start of screen memory at $D000 this value would be put in memory as 00 D0.
The initialisation routine is thus

```
INITIALISE LDA #$00
           STA TVPUT+1
           LDA #$D0
           STA TVPUT+2
```

The next operation would be to set up the accumulator with the blank character and do the first store:

```
           LDA #$20
TVPUT      STA $D000
```

Now comes a sequence which will increment the pointer. The first step is to increment the low byte.

```
           INC TVPUT+1
```

Next test to see if the Z flag is set to true (the location will overflow to $00 after 256 increments). If Z is not zero then loop back and do the next location

```
           BNE TVPUT
```

If Z was true then the high byte must be incremented

```
           INC TVPUT+2
```

Now a test is needed to see if all memory has been cleared. With the instructions available so far the easiest method is to use an index register as a counter to keep track of how many 'pages' of memory are left to be cleared. On a 1K screen there are 4 pages and a 2K screen has 8 pages (a page is 256 consecutive memory locations). The register is initially set to the number of pages of screen memory less one (remember that 0 will count as a number too). The count can be kept by decrementing it each time the high order byte of the pointer is incremented. If the x register is used the next step would be

```
           DEX
```

*Next page please*

**********

## LIBRARY NEWS

*Ron Kerry*

A firm in Queensland has offered to club members a 300 page Synertek catalogue and some application notes on the 65XX uP family. The catalogue is normally valued at $5.00, however the only charge to KAOS members is $1.20 to cover mailing costs. Also available a limited number of 'AIM' application notes. The address to write to ENERGY CONTROL. ATTN. MR KEN CURRY,

Would those members who borrowed magazines prior to the September meeting, please return them at the October meeting. A special amnesty period will apply with no fines for late returns. Remember the magazines have been donated by members for use by ALL members and loans are for a period of one month unless other arrangements have been made.

When the register contents become -1 ($FF) all memory has been cleared. While x is positive then, the program must loop back to blank the next memory location.

```
ENDTEST    BPL TVPUT
```

Once this branch test fails the program is finished.

All that remains now is to go back and put in the line to initialise the index register. The completed program becomes

```
INIT       LDA #$00
           STA TVPUT+1
           LDA #$D0
           STA TVPUT+2
           LDA #$20
           LDX #$07 ($04 for 1K screen)
TVPUT      STA $D000
           INC TVPUT+1
           BNE TVPUT
           INC TVPUT+2
           DEX
ENDTST     BPL TVPUT
           END
```

Because this program uses a self modifying code it must be located in RAM. It would be suitable to use in a disk based system, but is not really appropriate for a ROM based computer.

**********

## THE RABBLE OZI EXPANSION BOARD

*Greg Kilfoyle*

Recently I purchased a Rabble Ozi Board kit from COMP-SOFT. It took about 5 hours to build and about 3 hours to find 1 solder blotch and a few dry solder joints on the PSG's (Programmable Sound Generators). With the board being solder masked the risk of solder blotches was greatly reduced. The solder masking does create one problem though, some of the pin holes are slightly over-masked and there is very little contact exposed. For these holes an extra second with the soldering iron can avoid dry joint. I found an easy way to put the I.C. sockets in was to put a number of sockets in the board at once, place a folded towel (or foam) over them, and holding the towel against the sockets, put the board towel side down on the table. The flexibility of the towel holds the sockets evenly against the board, and two hands are left free for soldering. When placing discrete components on the board (resistors, capacitors etc.), I place a number in the board at once, bend the wires on the under side to hold them in place and solder the lot in one go.

After getting the board working, I immediately wanted to try out the PSG's. Not wanting to load the Assembler/Editor to try out the load program given with the Rabble board, I converted the program to BASIC for ease of use. Using data statements to hold the register values for the PSG, I ran the program. Hey Presto! A tone came blaring out of the little speaker. That was where the success stopped, any further loading of the PSG registers gave unpredictable results, so I wrote a program to read the PSG registers after the load program wrote to them, only to find that the registers weren't loading properly. After rewriting the program several times and cursing a few more times with no success, I decided to enter the original assembler load program and call it using the USR(X) function. This cured the problem, all register loads worked without error - the only conclusion I can draw from this is that BASIC is too slow to drive the PSG's, if anyone can clarify this for me I would be most grateful. Overall, I am very impressed with the Rabble board, for ease of building, capabilities available on a single board and at $305 for the kit, it is excellent value for money.

I will be writing a generalised 'MUSIC MAKER' program to drive the PSG's, enabling music scores to be entered with various sounds available for each voice, this will be released through KAOS at a later date.

Each of the FIELDS will occupy the next immediate block of characters after the previous FIELD. The starting point of each Field is called the INDEX. The special number is 10 characters long and is held in positions 0 thru 9, the next FIELD is held in positions 10 thru 35 and the next from 36 thru 39 and so on. I have tried to show this and the INDEX value of each point below. The different letter represents a different FIELD.

```
SnSnSnSnSnSnXXXXXXXXIIISSSSSSSSSSSSSSSTTTTTTTTTTTTTYYYPPPPPPPPPCCCCCCCCCCCCCCCCC next
0          10      25 28           53          73 76      85             135 record
```

If we want to input a particular FIELD all that we have to do is to set the INDEX of the Channel to the value of the INDEX of the start of our FIELD. and to get a particular FIELD from a particular RECORD, all we have to do is set the INDEX to the following formula.
INDEX X =FIELD LENGTH * RECORD NUMBER + SPECIAL NUMBER Character Total.
Remember that the 1st Record is RECORD 0, the 2nd is RECORD 1 and so on.
If all this is just too much for you try looking through the sample program section to see how it works.

FIND$ is a command to find a string of characters of up to 32 long in a particular Data file. It is used in the form INPUT"SEARCH STRING";S$ (from the keyboard) then the next program line is FINF S$,X where X=the channel. As mentioned, the search string can be up to 32 characters long and if you are not sure of the spelling, you can substitute '&' symbols for single or groups of letter or numbers. One very important point is to always set the INDEX of the channel back to a value below where the string is likely to be found. A good idea is to always set the index to 0, that is INDEX X =0.

I would be very interested in hearing from interested people who have 65U or if they have 8" DISK systems.

David Tasker,

```
 10  OPEN "FILNAM",1
 20  INDEX<1>=0
 30  INPUT"SEARCH STRING less than 32 Characters";S$
 40  FIND S$,1
 50  IFINDEX(1)>1E8THENPRINT"SEARCH STRING NOT FOUND":GOTO30
100  INDEX<1>=0
110  INPUT%1,N:REM N=NUMBER OF CHARACTERS STORED
120  E=10:REM START POINT OF IST FIELD
130  INDEX<1>=E
135  INPUT%1,SU$:REM SURNAME
140  INDEX<1>=E+25
145  INPUT%1,IN$:REM INITIALS
150  INDEX<1>=E+25+3
155  INPUT%1,ST$:REM STREET
160  INDEX<1>=E+25+3+25
165  INPUT%1,TN$:REM TOWNS
170  REM AND SO ON UNTIL THE END OF THE RECORD
171  REM NOW MAKE E =TO E +TOTAL FIELD LENGTH
175  E=E+135
176  REM NOW PRINT THEM OUT
178  PRINTSU$:PRINTIN$:PRINTST$:PRINTTN$ ETC ETC
180  REM GET ANOTHER RECORD
182  INPUT"ENTER A C TO CONTINUE";C$
183  GOTO130
184  :
185  CAN YOU WORK OUT HOW TO CREATE THE INITIAL FILE. Clues. N= the
186  total number of characters used and you PRINT%1,N and if
187  E gets to be greater than N then you have gone past the end of File
188  SET THE INDEX = to 0 to print N.  Increment the index at the same
189  rate as the fields increment and INPUT from the keyboard your
190  variables and then PRINTIT%1,VARIABLE$ AND SO ON.
195  GOOD LUCK
```

## THE MEETING WAS KAOS

*by Ron Cork*

Actually, the meetings are becoming less 'kaotic', there were only about 90 members at the last meeting. For an Annual-General Meeting, which it was, this is a poor turn-up. Being an annual-general meeting, all positions on the executive committee were automatically made vacant and open for new nominations. There were no new candidates so therefore, all current sitting?? principals were automatically re-elected un-opposed, without even a vote being taken. This is democracy at its best, the supreme vote of confidence in our leaders!!! It was also our second anniversary, an event of tremendous significance which went almost unnoticed, (like the anniversary cake, which was virtually invisible).

The biggest news of the day was the annoucement that MA-COM are disbanding the OSI branch of the company. This will result in all OSI personal computer users being left out in the cold. So what's new you say. I think that we, here in Australia in general and KAOS in particular, do more in both hardware and software development for the personal computer line than OSI anyway.

Ritchie Laird announced the amalgamation of Afig and the original Melbourne based group. Jeff Rae had his Acorn Atom on display. At $400 all-up, it comes with a 6502 mpu, hi-res and colour. A floppy controller and 4K dos-in-rom (which leaves the entire disk for data storage), is also available, at a price! Jeff also said he would arrange to have a demo soon of the Proton (whatever that is).

David Anear showed us a design for a logic probe that should enable anyone who hasn't got one, to get one, at very little cost (approx. $3). He also announced (as if we didn't already know) that the 6502 is the largest selling 8 bit mpu chip in the world, and with good reason. MOS are also now producing a new range of mpu's, viz: 65c02 - dual 8 bit (psuedo 16 bit), and others with inbuilt clocks at 2, 4, & 6 MHz. They have just released in the USA a true 16 bit chip as well.

Bill Chilcott is a little troubled by the slow? sales of the Rabble board. Times are tough Bill, and life wasn't meant to be easy. He is also working on a new addition to the line, an OSI (that word again) compatable system in kit form for an estimated cost of $250, with 32/32, 32/64, and 24/80 screen formats. With luck, Bill hopes to have a proto-type on display at the next meeting.

A new Assembler will be demonstrated by COMP-SOFT at the next meeting, it comes with macro instructions and will be available on cassette, disk or ROM, approx. cost $50. John Whitehead has another version of the disassembler, (BASIC), that will put in labels. Ron Kerry would like all those who borrowed magazines from the library, before he took over, to see him so that the records can be updated. This is most important, so be honest and get in touch with Ron asap.

The 48K boards should be here by the time you read this. Construction is even better than expected; thicker boards, heavier copper, and the component overlay is included. At 2 MHz it draws only 340 ma. Bare board price, to KAOS members is around $75, built and tested, around $350. A real bargain.

Now back to our serial. This episode of the saga of GTBUG finds Tony wallowing in the depths of self pity and guilt at the reluctance of THE BUG to manifest itself into the realms of the real world. In his vain but futile attempts to rid himself of this affliction, he has started pointing the finger at the original carrier of THE BUG, a certain G.N., whose name will remain secret on the grounds that it's revealment may tend to incinerate me. Stay tuned for more gripping? episodes. This article will self destruct in 5 seconds.

DEAR ~~ABBY~~ PAUL

*Your Questions Answered by Paul Dodd (Our Resident Expert)*

At the moment most of the questions are sent in by one or two people and sometimes I even have to resort to inventing a few myself, so keep the questions rolling in. I don't mind how complicated (or simple) the questions are - if I can't answer them, I will find someone who can. Simple problems are sometimes more frustrating than the more complex ones. If enough people send letters, I will change the format of the article and print whole letters or excerpts from them, and a full answer instead of this current Q/A arrangement.

Q. I will recently purchased a Tasan Video board as a kit from COMP-SOFT, it works very well except that spurious characters occasionally appear. What causes this problem?
A. The answer to this problem is simple. This is the famous Read/Write problem with the Tasan board. For a full description, and answer see KAOS Vol.2 No.11.

Q. What is the 'Garbage Collection Problem' in OSI BASIC, and how can it be fixed?
A. If you use strings in OSI BASIC, they take up room from the top of memory downwards, each time you re-use a string, BASIC checks to see if it is shorter than the existing string, if it is, then the new string will fit in the old position and the excess area is marked as garbage. If the new string is too long the old position is marked as garbage, and the new string is tacked onto the end of the string area. When the string area meets up with a variable area, BASIC decides to clean up the string area. All the strings in memory are shift up to fill in the blank (garbage) areas. This process is called garbage collection, and it is this that doesn't work on the OSI (the problem also shows up on the PET and the APPLE to name just two). There are a few tapes around (including Aardvarks 'Adventure Setup' tapes) which will fix the problem, but most will run only in 8K. The best solution is to purchase a replacement BASIC ROM (ROM 3), I believe that John Whitehead sells these for some minimal amount.
NOTE: The problem does not occur in Disk BASIC.

Q. I have just built another Tasker RAM card to plug into my Mother board, but when I plug it in, and turn the computer on, my computer fails to respond to 'BREAK', can you help me?
A. The RAM card for the Tasker Buss is particularly easy to debug as all the address, control, and data lines are buffered. In all probability you have an address line shorted on the computer side of the address buffers - look carefully with a magnifying glass.

**********

SUPERBOARD (QLD) USER GROUP MEETING 26/9/82
*by Ed Richardson*

Attendance 21 Computers 7

Brendan Vowles had the Tasan board fitted and working, and showed the comprehensive installation and instruction manual. The display was very stable and the smaller characters made the font look so much better, and a list of a program was so much easier to read. The board was very good quality and required only one amp when fully populated.

Bernie Wills demonstrated the various character sets on his machine. Bernie also bought in his Eprom programmer and demonstrated the Basic that he had written to drive it. Members showed considerable interest in Bernie's various neat interfaces.
Stephen Border offered to provide a listing of any program on his new Tandy Model VII printer.
Bob Best had typed a number of interesting programs into his C4P and many were suitable for both C1 and C4, Bob was kept busy.

An announcement that some old electro-mechanical printers, complete with power supplies, were to be free for the taking on Tuesday was greeted with lukewarm interest. (When the time came, however, a large and assorted range of vehicles turned up to receive the booty.)

The meeting welcomed a number of new members, including students and teachers from some of the schools which have joined up. As usual, the meeting turned into a program swapping exercise at the end. The meeting closed at 5.30pm, when I heaved the last two enthusiasts out the door. Next meeting in a couple of months, with luck.

**********

## CMOS 48K STATIC RAM BOARD

The new Australian designed and manufactured 48K static CMOS RAM board is now available. This board was primarily designed as an expansion or replacement board for OSI Model C2-4P, C2-OEM, C3, C4 and C8p machines. The board will replace the 520, 527, 530 and 535 memory boards and has the following features:-

a/ Low power CMOS RAM (2K X 8 6116 static RAM) uses less power than 8K 2114 RAM - 5V at 300-400mA.
b/ 2 MHz operation guaranteed
c/ 20 address lines for multi-user or memory management systems with decoding to put the board in any of 16 pages.
d/ Strapable for various other configurations
e/ Suitable for C1P, S/Board or UK101 with the addition of an adaptor board.

This board is a double sided, through hole plated printed board with solder resist coating and component overlay and comes with a manual detailing options etc.

| SPECIAL PRICES TO KAOS MEMBERS | |
|---|---|
| Bare board only with manual | $ 75.00 |
| Partial Kit (no RAM) | $105.00 |
| Complete Kit | $275.00 |
| Assem. and Tested, with guarantee | $350.00 |

To purchase or for more information contact:-
COMP-SOFT
David Anear (DMA Computing)
Nigel Bissett

**********

## SYM-POSIUM

*Brian Campbell*

As promised last month, here is the table of VIA timer values required to produce the correct frequencies of the musical scale. The reference tone A at 440 Hz is underlined.

Also, the circuit for the two VIA mixer and the digital-analogue converter is given. Please note that DAC circuit has a high output impedance and requires either a high input impedance amplifier or an op-amp buffer. The input of the op-amp can be connected directly to pin 4 of the IC. The op-amp must be used as an inverting amplifier with the feedback resistor only.

You should now be able to create your own SYM-phony orchestra.

WANTED: One article for next month. (1 - 2 pages). New or used OK. If magazine model, it must be fully reconditioned to suit. No reasonable offer refused.

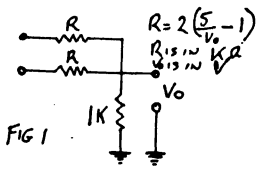


FIG 1

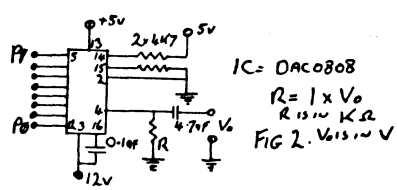


FIG 2.

MUSICAL TONE FREQUENCY TABLE

After initializing the 6522, enter the last number given in the table below into $AX04. When the other number is entered into $AX05 the tone will begin.

| Note | POKE | Value | Note | POKE | Value |
|---|---|---|---|---|---|
| A# | 268 | 36 | | | |
| B | 253 | 23 | | | |
| C | 238 | 226 | C# | 225 | 122 |
| D | 212 | 210 | D# | 200 | 224 |
| E | 189 | 154 | | | |
| F | 178 | 246 | F# | 168 | 234 |
| G | 159 | 111 | G# | 150 | 124 |
| A | 142 | 10 | A# | 134 | 17 |
| B | 126 | 139 | | | |
| C | 119 | 112 | C# | 112 | 188 |
| D | 106 | 104 | D# | 100 | 111 |
| E | 94 | 204 | | | |
| F | 89 | 122 | F# | 84 | 116 |
| G | 79 | 183 | G# | 75 | 61 |
| A | 71 | 4 | A# | 67 | 8 |
| B | 63 | 68 | | | |
| C | 59 | 183 | C# | 56 | 93 |
| D | 53 | 51 | D# | 50 | 55 |
| E | 47 | 101 | | | |
| F | 44 | 188 | F# | 42 | 57 |
| G | 39 | 218 | G# | 37 | 158 |
| A | 35 | 129 | A# | 33 | 131 |
| B | 31 | 161 | | | |

| Note | POKE | Value | Note | POKE | Value |
|---|---|---|---|---|---|
| C | 29 | 219 | C# | 28 | 46 |
| D | 26 | 153 | D# | 25 | 26 |
| E | 23 | 178 | | | |
| F | 22 | 93 | F# | 21 | 28 |
| G | 19 | 236 | G# | 18 | 206 |
| A | 17 | 192 | A# | 16 | 193 |
| B | 15 | 208 | | | |
| C | 14 | 237 | C# | 14 | 22 |
| D | 13 | 75 | D# | 12 | 140 |
| E | 11 | 216 | | | |
| F | 11 | 46 | F# | 10 | 141 |
| G | 9 | 245 | G# | 9 | 102 |
| A | 8 | 223 | A# | 8 | 95 |
| B | 7 | 231 | | | |
| C | 7 | 117 | C# | 7 | 10 |
| D | 6 | 165 | D# | 6 | 69 |
| E | 5 | 235 | | | |
| F | 5 | 150 | F# | 5 | 70 |
| G | 4 | 250 | G# | 4 | 178 |
| A 440 | 4 | 111 | A# | 4 | 47 |
| B | 3 | 243 | | | |
| C | 3 | 186 | C# | 3 | 132 |
| D | 3 | 82 | D# | 3 | 34 |
| E | 2 | 245 | | | |
| F | 2 | 202 | F# | 2 | 162 |
| G | 2 | 124 | G# | 02 | 88 |
| A | 2 | 54 | A# | 2 | 23 |
| B | 1 | 248 | | | |
| C | 1 | 220 | C# | 1 | 193 |
| D | 1 | 168 | D# | 1 | 144 |
| E | 1 | 121 | | | |
| F | 1 | 100 | F# | 1 | 80 |
| G | 1 | 61 | G# | 1 | 43 |
| A | 1 | 26 | A# | 1 | 10 |
| B | 0 | 251 | | | |

| Note | POKE | Value | Note | POKE | Value |
|---|---|---|---|---|---|
| C | 0 | 237 | C# | 0 | 224 |
| D | 0 | 211 | D# | 0 | 199 |
| E | 0 | 188 | | | |
| F | 0 | 177 | F# | 0 | 167 |
| G | 0 | 158 | G# | 0 | 149 |
| A | 0 | 140 | A# | 0 | 132 |
| B | 0 | 125 | | | |
| C | 0 | 118 | C# | 0 | 111 |
| D | 0 | 105 | D# | 0 | 99 |
| E | 0 | 93 | | | |
| F | 0 | 88 | F# | 0 | 83 |
| G | 0 | 78 | G# | 0 | 73 |
| A | 0 | 69 | A# | 0 | 65 |
| B | 0 | 62 | | | |
| C | 0 | 58 | C# | 0 | 55 |
| D | 0 | 51 | D# | 0 | 48 |
| E | 0 | 46 | | | |
| F | 0 | 43 | F# | 0 | 40 |
| G | 0 | 38 | G# | 0 | 36 |
| A | 0 | 34 | A# | 0 | 32 |
| B | 0 | 30 | | | |
| C | 0 | 28 | C# | 0 | 26 |
| D | 0 | 25 | D# | 0 | 23 |
| E | 0 | 22 | | | |
| F | 0 | 21 | F# | 0 | 19 |
| G | 0 | 18 | G# | 0 | 17 |
| A | 0 | 16 | A# | 0 | 15 |
| B | 0 | 14 | | | |
| C | 0 | 13 | C# | 0 | 12 |
| D | 0 | 12 | D# | 0 | 11 |
| E | 0 | 10 | | | |

**********

Members of KAOS living in Sth Australia may be interested in meetings of a local User Group, 6502 SA. The next meeting is on November 13th, and you can contact Mark Holderness, 35 Hawthorndene Drive, Glenalta, SA 5052 for more details. At the moment the majority of 6502 SA members own OSI's.

**********

FOR SALE

COMPUKIT UK101 MICRO with 8K RAM, New Monitor, Editor/Assembler, Disassembler, 300/600 Baud, Cassettes, Manuals etc., in case with external power supply $250.00. Ring John Button

MODEL 15 PRINTER G.C., Tape distributer 6 channel G.C., Re-perforator as is, Teleprinter Tester-generates RYs or code as required with CRO in metal case. The lot $70.00. Contact Doug Johnson, [illegible]

ASR 33 TTY including paper tape punch and reader. Half or full Duplex; very good condition $200.00. Contact Ross

Challenger 1P metal case with 5A power supply fitted $80.00 ONO ph.568 6273

Two boards compatable with the Rockwell AIM-65. 1. Motorola 68R2M Video board = 6845 chip, PIA, 4K own RAM, fully programmable $180.00. 2. BETA 6502DM 32K dynamic memory board, suits any 8 bit computer $135.00. SHUGART SA400 disk drives at $210.00 each or as two units mounted in cabinet with power supply at $480.00. Controller card for above, WD1771 chip, by SWTP, $95.00. All equipment is fully documented. Contact Mr Fausto Teixeira,